David Kirby

ECE 595: Advanced Technical Cybersecurity

Spring, 2022

Capture the Flag

The first part of this three-part reverse-engineering exercise was to determine the function name using the nm or readelf commands (see Figure 1). The next part was to take a look at the disassembled code in GDB (with the pwndbg interface) to determine the purpose of the function (see Figure 2). We can see that the assembly code uses two C library functions – `snprint` and `strncat`, along with a series of jumps and moves. If we compare this to the disassembled code of our previous assignment, we see that the use of jumps and moves is similar to the ones we saw in our for and while loops. This lead me to believe that we were dealing with a loop function. It turns out that running through our `main` function, and setting breakpoints at the called functions, we see that the program is performing an iterative loop that multiplies i by i and prints the result. It goes through this iteration five times, with the result of each iteration then concatenated with the string `_msg` as shown below. The final contents of the buffer at program termination are `16_msg`.

Output of five iterations of the function.

```
0
0_msg
1
1_msg
4
4_msg
9
9_msg
16
16_msg
```

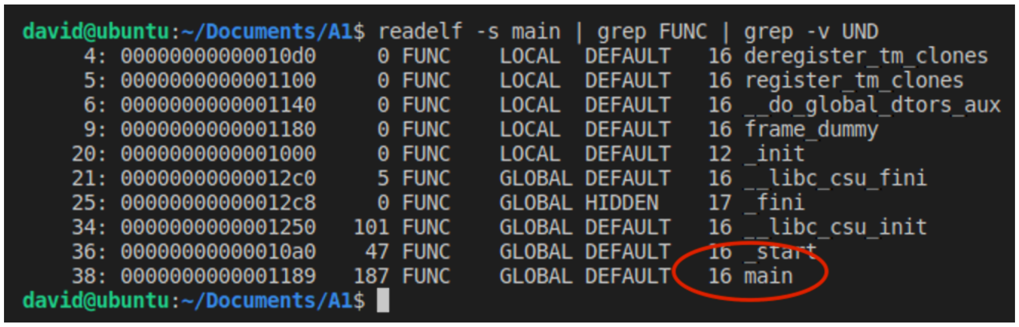

```
david@ubuntu:~/Documents/A1$ readelf -s main | grep FUNC | grep -v UND
     4: 00000000000010d0     0 FUNC    LOCAL  DEFAULT   16 deregister_tm_clones
     5: 0000000000001100     0 FUNC    LOCAL  DEFAULT   16 register_tm_clones
     6: 0000000000001140     0 FUNC    LOCAL  DEFAULT   16 __do_global_dtors_aux
     9: 0000000000001180     0 FUNC    LOCAL  DEFAULT   16 frame_dummy
    20: 0000000000001000     0 FUNC    LOCAL  DEFAULT   12 _init
    21: 00000000000012c0     5 FUNC    GLOBAL DEFAULT   16 __libc_csu_fini
    25: 00000000000012c8     0 FUNC    GLOBAL HIDDEN    17 _fini
    34: 0000000000001250   101 FUNC    GLOBAL DEFAULT   16 __libc_csu_init
    36: 00000000000010a0    47 FUNC    GLOBAL DEFAULT   16 _start
    38: 0000000000001189   187 FUNC    GLOBAL DEFAULT   16 main
david@ubuntu:~/Documents/A1$
```

FIGURE 1: DETERMINING THE FUNCTION NAME USING READELF.

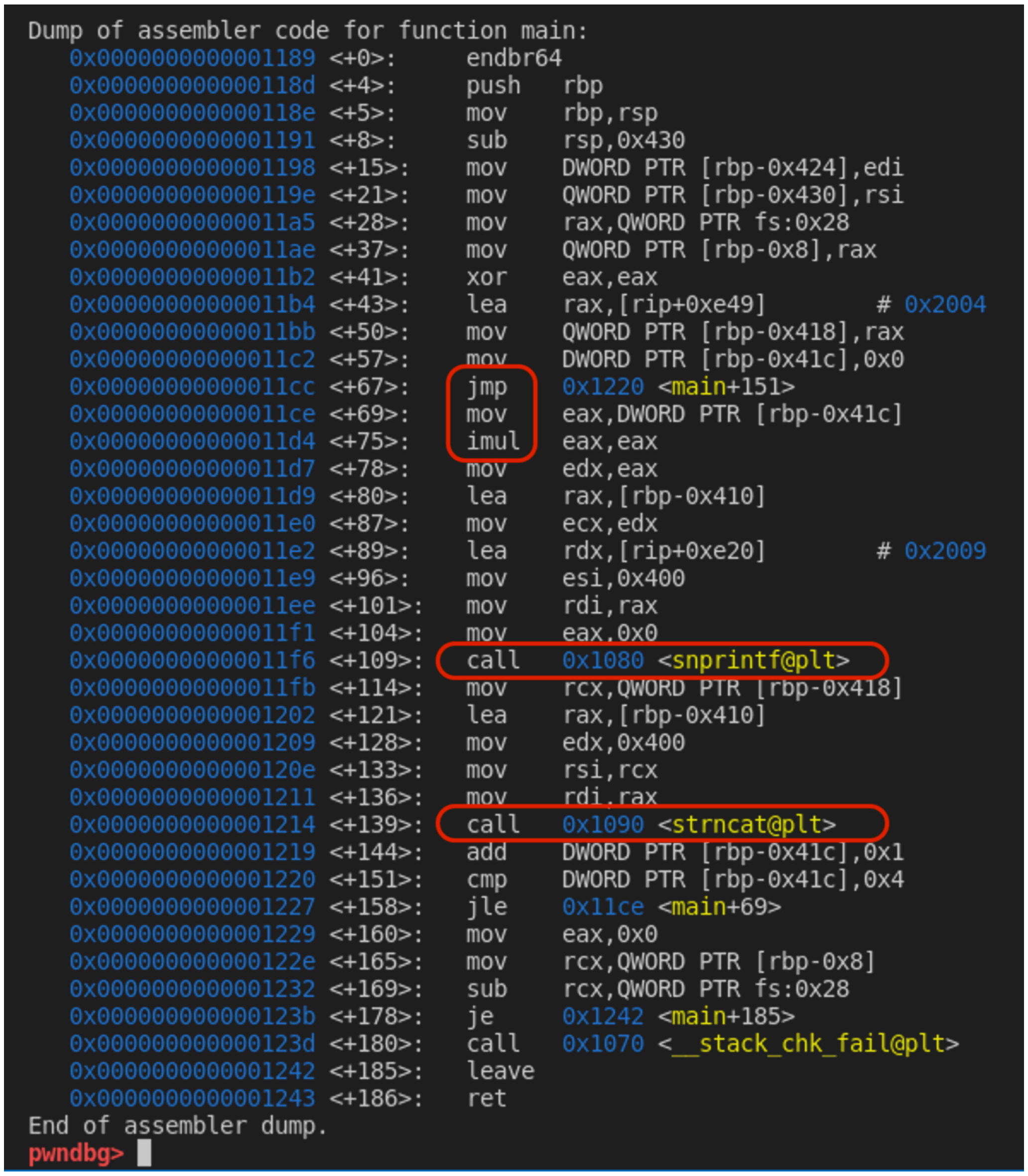
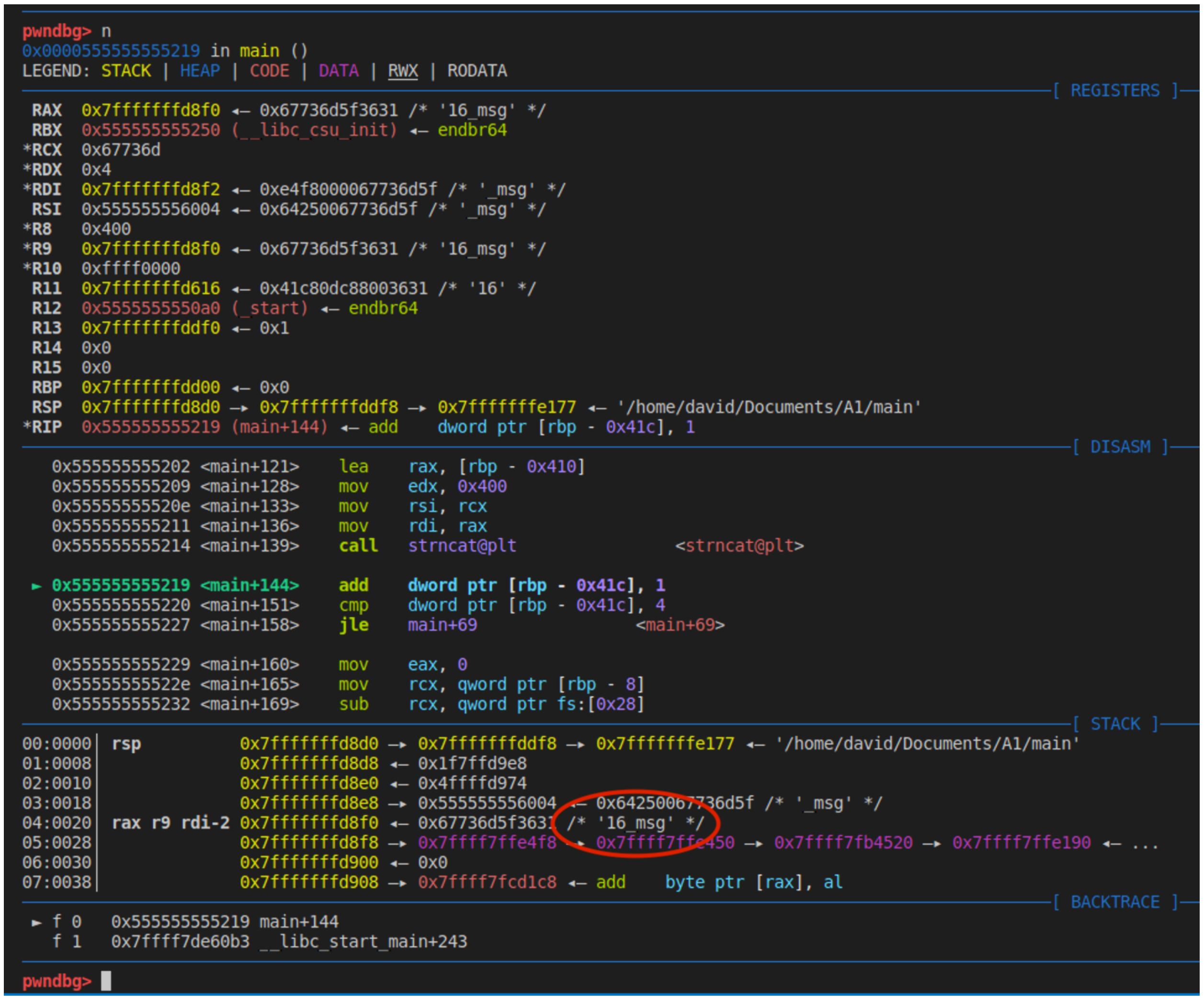

```
Dump of assembler code for function main:
   0x0000000000001189 <+0>:     endbr64
   0x000000000000118d <+4>:     push   rbp
   0x000000000000118e <+5>:     mov    rbp,rsp
   0x0000000000001191 <+8>:     sub    rsp,0x430
   0x0000000000001198 <+15>:    mov    DWORD PTR [rbp-0x424],edi
   0x000000000000119e <+21>:    mov    QWORD PTR [rbp-0x430],rsi
   0x00000000000011a5 <+28>:    mov    rax,QWORD PTR fs:0x28
   0x00000000000011ae <+37>:    mov    QWORD PTR [rbp-0x8],rax
   0x00000000000011b2 <+41>:    xor    eax,eax
   0x00000000000011b4 <+43>:    lea    rax,[rip+0xe49]        # 0x2004
   0x00000000000011bb <+50>:    mov    QWORD PTR [rbp-0x418],rax
   0x00000000000011c2 <+57>:    mov    DWORD PTR [rbp-0x41c],0x0
   0x00000000000011cc <+67>:    jmp    0x1220 <main+151>
   0x00000000000011ce <+69>:    mov    eax,DWORD PTR [rbp-0x41c]
   0x00000000000011d4 <+75>:    imul   eax,eax
   0x00000000000011d7 <+78>:    mov    edx,eax
   0x00000000000011d9 <+80>:    lea    rax,[rbp-0x410]
   0x00000000000011e0 <+87>:    mov    ecx,edx
   0x00000000000011e2 <+89>:    lea    rdx,[rip+0xe20]        # 0x2009
   0x00000000000011e9 <+96>:    mov    esi,0x400
   0x00000000000011ee <+101>:   mov    rdi,rax
   0x00000000000011f1 <+104>:   mov    eax,0x0
   0x00000000000011f6 <+109>:   call   0x1080 <snprintf@plt>
   0x00000000000011fb <+114>:   mov    rcx,QWORD PTR [rbp-0x418]
   0x0000000000001202 <+121>:   lea    rax,[rbp-0x410]
   0x0000000000001209 <+128>:   mov    edx,0x400
   0x000000000000120e <+133>:   mov    rsi,rcx
   0x0000000000001211 <+136>:   mov    rdi,rax
   0x0000000000001214 <+139>:   call   0x1090 <strncat@plt>
   0x0000000000001219 <+144>:   add    DWORD PTR [rbp-0x41c],0x1
   0x0000000000001220 <+151>:   cmp    DWORD PTR [rbp-0x41c],0x4
   0x0000000000001227 <+158>:   jle    0x11ce <main+69>
   0x0000000000001229 <+160>:   mov    eax,0x0
   0x000000000000122e <+165>:   mov    rcx,QWORD PTR [rbp-0x8]
   0x0000000000001232 <+169>:   sub    rcx,QWORD PTR fs:0x28
   0x000000000000123b <+178>:   je     0x1242 <main+185>
   0x000000000000123d <+180>:   call   0x1070 <__stack_chk_fail@plt>
   0x0000000000001242 <+185>:   leave
   0x0000000000001243 <+186>:   ret
End of assembler dump.
pwndbg>
```

FIGURE 2: DISASSEMBLED CODE USING GDB.



```
pwndbg> n
0x0000555555555219 in main ()
LEGEND: STACK | HEAP | CODE | DATA | RWX | RODATA
─────────────────────────────────────────────────────────────[ REGISTERS ]─
 RAX  0x7fffffffd8f0 ◂— 0x67736d5f3631 /* '16_msg' */
 RBX  0x555555555250 (__libc_csu_init) ◂— endbr64
*RCX  0x67736d
*RDX  0x4
*RDI  0x7fffffffd8f2 ◂— 0xe4f8000067736d5f /* '_msg' */
 RSI  0x555555556004 ◂— 0x64250067736d5f /* '_msg' */
*R8   0x400
*R9   0x7fffffffd8f0 ◂— 0x67736d5f3631 /* '16_msg' */
*R10  0xffff0000
 R11  0x7fffffffd616 ◂— 0x41c80dc88003631 /* '16' */
 R12  0x5555555550a0 (_start) ◂— endbr64
 R13  0x7fffffffddf0 ◂— 0x1
 R14  0x0
 R15  0x0
 RBP  0x7fffffffdd00 ◂— 0x0
 RSP  0x7fffffffd8d0 —▸ 0x7fffffffddf8 —▸ 0x7fffffffe177 ◂— '/home/david/Documents/A1/main'
*RIP  0x555555555219 (main+144) ◂— add    dword ptr [rbp - 0x41c], 1
───────────────────────────────────────────────────────────────[ DISASM ]─
   0x555555555202 <main+121>    lea    rax, [rbp - 0x410]
   0x555555555209 <main+128>    mov    edx, 0x400
   0x55555555520e <main+133>    mov    rsi, rcx
   0x555555555211 <main+136>    mov    rdi, rax
   0x555555555214 <main+139>    call   strncat@plt                <strncat@plt>

 ► 0x555555555219 <main+144>    add    dword ptr [rbp - 0x41c], 1
   0x555555555220 <main+151>    cmp    dword ptr [rbp - 0x41c], 4
   0x555555555227 <main+158>    jle    main+69                <main+69>

   0x555555555229 <main+160>    mov    eax, 0
   0x55555555522e <main+165>    mov    rcx, qword ptr [rbp - 8]
   0x555555555232 <main+169>    sub    rcx, qword ptr fs:[0x28]
────────────────────────────────────────────────────────────────[ STACK ]─
00:0000│ rsp         0x7fffffffd8d0 —▸ 0x7fffffffddf8 —▸ 0x7fffffffe177 ◂— '/home/david/Documents/A1/main'
01:0008│             0x7fffffffd8d8 ◂— 0x1f7ffd9e8
02:0010│             0x7fffffffd8e0 ◂— 0x4ffffd974
03:0018│             0x7fffffffd8e8 —▸ 0x555555556004 ◂— 0x64250067736d5f /* '_msg' */
04:0020│ rax r9 rdi-2 0x7fffffffd8f0 ◂— 0x67736d5f3631 /* '16_msg' */
05:0028│             0x7fffffffd8f8 —▸ 0x7ffff7ffe4f8 —▸ 0x7ffff7ffe450 —▸ 0x7ffff7fb4520 —▸ 0x7ffff7ffe190 ◂— ...
06:0030│             0x7fffffffd900 ◂— 0x0
07:0038│             0x7fffffffd908 —▸ 0x7ffff7fcd1c8 ◂— add    byte ptr [rax], al
────────────────────────────────────────────────────────────[ BACKTRACE ]─
 ► f 0   0x555555555219 main+144
   f 1   0x7ffff7de60b3 __libc_start_main+243
──────────────────────────────────────────────────────────────────────────
pwndbg> 
```

FIGURE 3: STACK SHOWING CONCATENATED MESSAGE.